1. A method for delivering applications over a network in which the business logic of the application is running on the backend server, the user interface of the application is rendered on a client-device who is connected to the backend server via a network . The Graphics User Interface API and event processing API of the application is implemented

5    to be network-aware instead of being local machine centric as traditional GUI APIs:

running an application on the backend server. The application in turn invokes GUI API

to present its user interface. However, the network-aware GUI API is invoked;

translating the application's presentation layer information into a pre-determined

10    format based messages which describes the Graphical User Interface, event processing registries and other related information. Such information describes the presentation layer of the application in a high level, object level, which minimizes network traffic;

sending such messages to the client device via a network;

processing the messages and rendering the user interface by a client-side program,

15    which delivers the best possible user experience for that device according to the capability of the specific client device.

transmitting necessary user input and client-side events back to the server by the client-side program via a predetermined protocol;

processing the user input and client-side events on the backend server, translating

20    such events and inputs as if they were locally generated, and sending such translated events and inputs to the application for processing;

encoding and routing the output of the application to the client device using the predetermined messaging format; and,

further processing the output by the client-side program to refresh the Graphical

25    User Interface thereat.


2. The method of Claim 1, wherein Graphics User Interface API and event processing API is Java Foundation Classes (including Swing, AWT and so on);


30    3. The method of Claim 1, wherein the client-side program is a computer program based on Operating System's API, such as Windows API, X Windows API and so on;


73

4. The method of Claim 1, wherein the client-side program is a wireless device program written using the device's Operating System's API, such as Palm API and Windows CE API;

5. The method of Claim 1, wherein the client-side program is Java program written using Java API;

6. The method of Claim 5, wherein the JAVA API is AWT, Personal Java, Java 2 Micro Edition based GUI API or Java Swing;

7. The method of Claim 1, wherein the predetermined protocol is HTTP.

8. The method of Claim 1, wherein the predetermined protocol is HTTPS.

9. The method of Claim 1, wherein predetermined protocol is WAP.

10. The method of Claim 1, wherein predetermined protocol is proprietary.

11. The method of Claim 1, wherein the predetermined messaging format is based on XML;

12. The method of Claim 1, wherein the predetermined messaging format is proprietary;

13. The method of Claim 1, wherein the network is the Internet.

14. The method of Claim 1, wherein the network is a local area network.

15. The method of Claim 8, wherein the local area network is a bandwidth-limited slow speed network.

16. The method of Claim 1, wherein the network includes a wireless network.

17. The method of Claim 11, wherein the client device is selected from the group consisting of workstations, desktops, laptops, PDAs, wireless devices and other edge devices;

5

18. The method of Claim 1, wherein the server and the client device are combined into one entity.

19. A server-side API based programming model for network programming, which frees

10 or greatly simplifies the complexity of network programming by freeing developers from client-side issues:

The presentation layer of the application is written using this server-side API;

The business logic layer and data layer of the application is written using other appropriate server-side technologies;

15 The supporting infrastructure of this server-side API sends the application' user interface information to the client-side device for presentation, handles communications problems, renders the application's user interface and dispatches necessary user input events back to the server for processing.

20 20. A method and system for delivering existing Java applications over the network without modification of the application's code and without downloading the application to the client side:

The system re-implements standard Java GUI APIs such as AWT and Swing into a network-aware implementation without changing the APIs, enabling existing Java

25 applications to run on this network-aware GUI API without modifications;

The Java application runs completely on the server-side. The network-aware API translates and delivers the application's presentation information into short messages based on formats such as XML via a certain communication protocol;

The system's client-side program that understands these messages interprets and

30 renders the user interface of the Java applications, essentially produces the look and feel of the application as if the entire application is running on the client device; The

75

client program further interacts with the user , dynamically updates the user interface and sends necessary user inputs back to server for processing;

The system's server program receives such user inputs, translates them into Java compatible user inputs, such as Java events, and further routes such user inputs to the Java application for processing;

The output of the Java application's processing is sent to the system's client program, which updates the user interface of the application.

21. A method and system for delivering the same application over some network to multiple devices, maximizing the user experience of each device by best leveraging the specific capability of each device, without rewriting the application specifically for each device:

The system runs the application on the server side;

The system's server-side program translates and delivers the presentation information of the application into messages based on selected format such as XML. Such messages contain high level description of the application's user interface. Such high level, instead of pixel level or graphics primitive level description, gives sufficient flexibility in interpretation without losing the gist of the information;

Specific client-side programs are built for each specific client device leveraging the special features of each device. This client-side program interprets these messages and renders the user interface of the applications in a way that is best optimized for the client device, delivering the best user experience possible on that specific client device.

The client-side program accepts user inputs, update the user interface, and sends necessary user inputs back to the server;

The system's server program receives such user inputs, translates them into application compatible user inputs, and further routes such user inputs to the application for processing;

The output of the application's processing is sent to the system's client program, which updates the user interface of the application accordingly.